EE 440 Homework #5

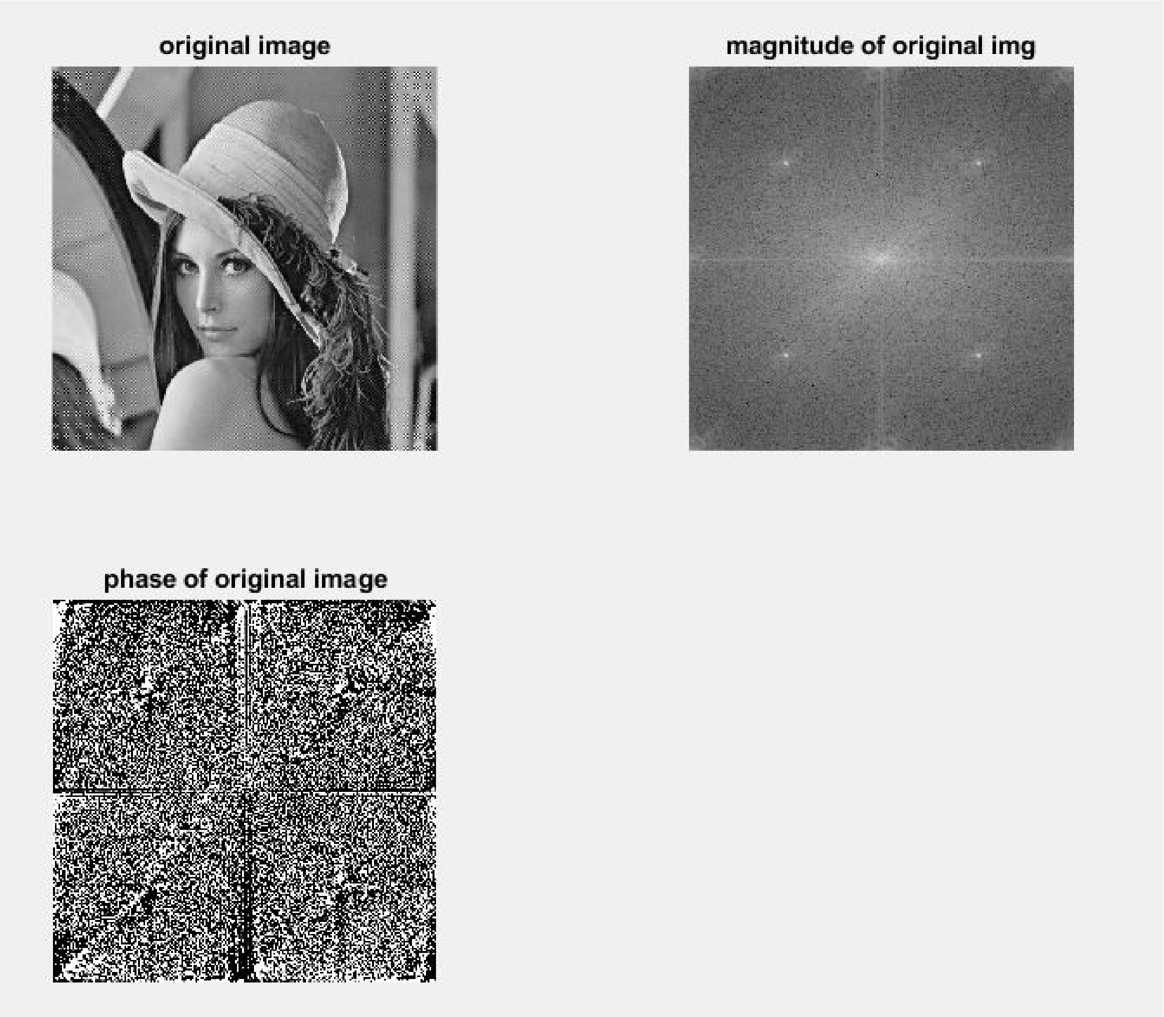
(Luke Jiang, 1560831)

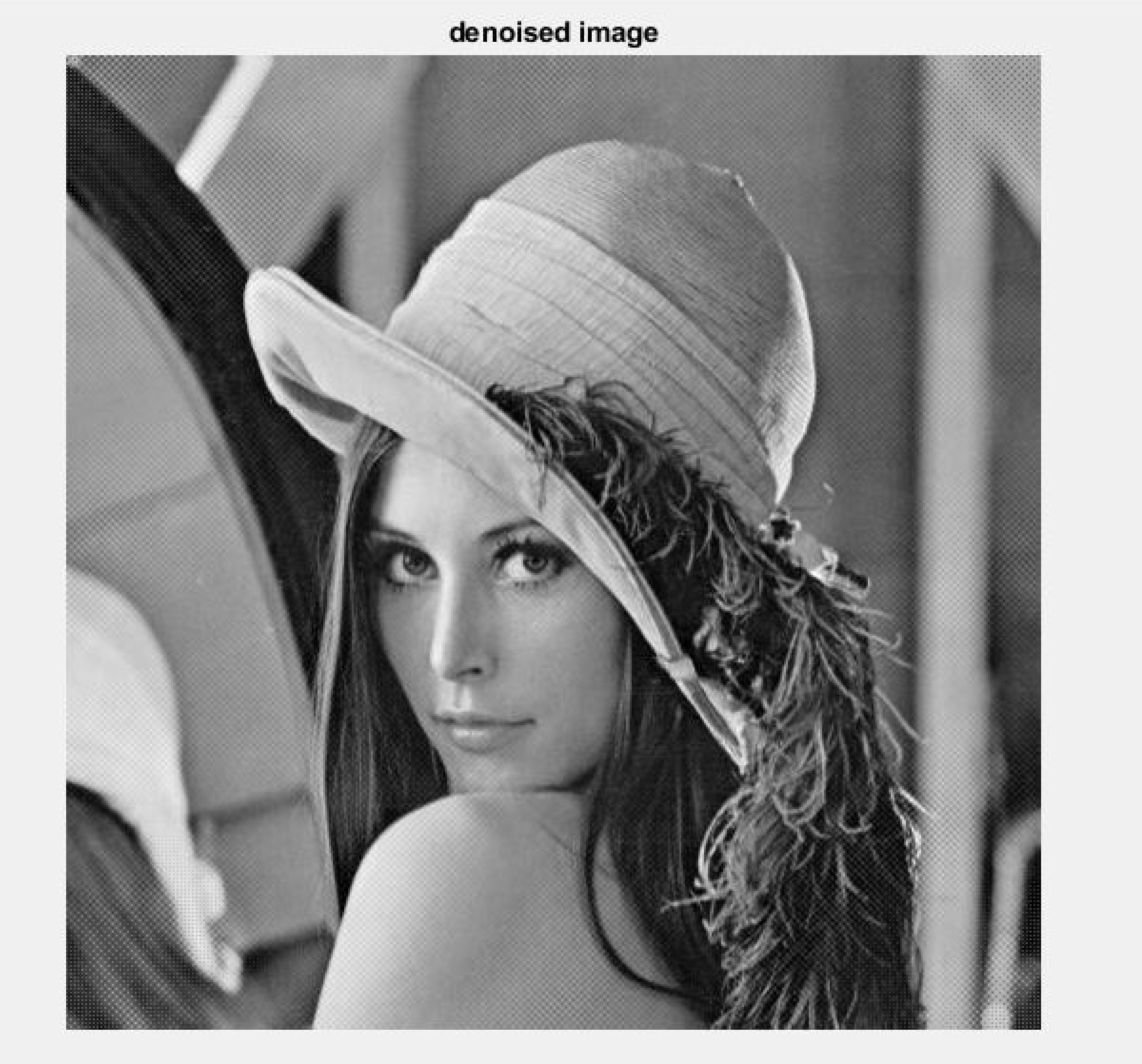
After reading the image, I performed FFT to the noised image as the hint suggests. I used data cursor to locate the position of four noise points on the magnitude plot. It turns out that zeroing one pixel only is insufficient for producing a discernible enhancement. Therefore, I also zeroed out several pixels around the noise points. I then used this denoised magnitude and the original phase to reconstruct the image in frequency domain. According to the definitions of magnitude and phase, the reconstructed denoised image can be calculated as:

$$F_{reconstructed} = magnitude_{modified} * e^{i*phase}$$

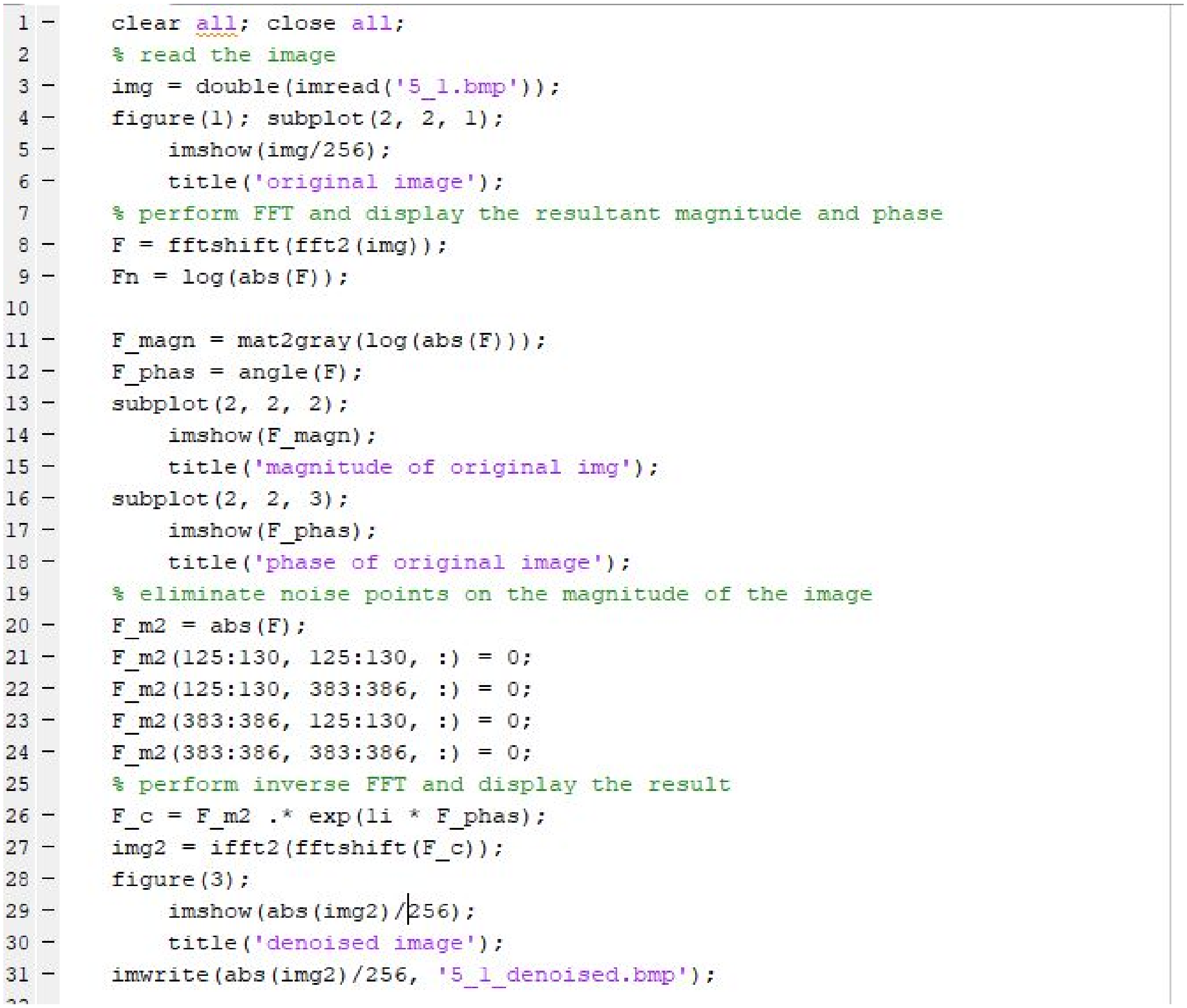
Since we performed fftshift( ) after using FFT, I then performed fftshift() to F_reconstructed before calling ifft2().

The results are the following:





The source code is the following:



```matlab
clear all; close all;
% read the image
img = double(imread('5_1.bmp'));
figure(1); subplot(2, 2, 1);
    imshow(img/256);
    title('original image');
% perform FFT and display the resultant magnitude and phase
F = fftshift(fft2(img));
Fn = log(abs(F));

F_magn = mat2gray(log(abs(F)));
F_phas = angle(F);
subplot(2, 2, 2);
    imshow(F_magn);
    title('magnitude of original img');
subplot(2, 2, 3);
    imshow(F_phas);
    title('phase of original image');
% eliminate noise points on the magnitude of the image
F_m2 = abs(F);
F_m2(125:130, 125:130, :) = 0;
F_m2(125:130, 383:386, :) = 0;
F_m2(383:386, 125:130, :) = 0;
F_m2(383:386, 383:386, :) = 0;
% perform inverse FFT and display the result
F_c = F_m2 .* exp(1i * F_phas);
img2 = ifft2(fftshift(F_c));
figure(3);
    imshow(abs(img2)/256);
    title('denoised image');
imwrite(abs(img2)/256, '5_1_denoised.bmp');
```